WHAT IS CLAIMED IS:

1. A method of operating a transaction system which supports bulk delegations of locks from one or more delegator transactions to one or more delegatee transaction, the method comprising:

validating at least a subset of the bulk lock delegations by, for each delegated lock owned by a delegator transaction, testing validity of delegating the lock based, at least in part, on ignore conflicts relationships amongst the delegatee transactions and between the delegatee transactions and otherwise incompatible-mode owners of locks, which would remain after completion of the bulk delegation.

2. A method, as recited in claim 1, further comprising:

performing the validated bulk lock delegations.

3. A method, as recited in claim 1, further comprising:

representing locks of identical value as references to a shared lock state encoding the value.

4. A method, as recited in claim 3, wherein a subset of the shared lock states is encoded in an associative table of shared lock states (TSLS).

5. A method, as recited in claim 1, further comprising:

obviating the testing of each lock for at least another subset of the bulk lock delegations by performing one or more conservative tests.

6. A method, as recited in claim 5,

wherein the one or more conservative tests include testing for disjunction of a set of delegator transactions and a *wset*.

7. A method, as recited in claim 6, further comprising:

adding a corresponding entry to the *wset* coincident with recording of a lock owned in a write mode; and

recomputing the *wset* by scanning the locks.

8. A method, as recited in claim 5, wherein the one or more conservative tests includes:

determining for each particular one of the delegator transactions, whether there are no delegatee transactions such that, for each type of conflict, a corresponding first set of transactions with which the delegatee can ignore conflicts is not a super-set of a second set of transactions that includes those transactions with which the particular delegator transaction can ignore conflicts of such type, excluding the delegator transactions themselves, and

if there is at least one write lock owned by at least one of the delegator transactions, then determining whether all delegatee can ignore conflicts with each other.

9. A method, as recited in claim 1, performed by a lock manager in a transaction processing system.

10. In a computational system wherein a value of a lock is encoded to identify (i) a set of one or more transactions that own the lock and (ii) respective one or more modes in which such transactions own the lock, and wherein at least some locks of equal value are represented using a same shared lock state, a method of validating bulk delegation of locks from one or more delegator transactions to one or more delegatee transactions, the method comprising:

for at least a subset of the bulk delegations, and for each shared lock state encoding having an associated owner set that includes at least one of the delegator transactions, testing validity of delegating a lock represented thereby based, at least in part, on ignore conflicts relationships amongst

the delegatee transactions and between the delegatee transactions and otherwise incompatible-mode owners that would remain after completion of the bulk delegation.

11. A method, as recited in claim 10, further comprising:

for each of the bulk delegations, performing one or more conservative validation tests and thereby obviating, for another subset of the bulk delegations, the validity testing for each shared lock state encoding.

12. A method, as recited in claim 11,

wherein the one or more conservative tests include testing for disjunction of a set of delegator transactions and a *wset*.

13. A method, as recited in claim 12, further comprising:

adding a corresponding entry to the *wset* coincident with recording of a lock value held in a write mode; and

recomputing the *wset* upon each scan of the shared lock states.

14. A method, as recited in claim 11, wherein the one or more conservative tests includes:

determining for each particular one of the delegator transactions, whether there are no delegatee transactions such that, for each type of conflict, a corresponding first set of transactions with which the delegatee can ignore conflicts is not a super-set of a second set of transactions that includes those transactions with which the particular delegator transaction can ignore conflicts of such type, excluding the delegator transactions themselves, and

if there is at least one write lock owned by at least one of the delegator transactions, then determining whether all delegatee can ignore conflicts with each other.

15. A method, as recited in claim 10,

wherein the shared lock states are at least partially encoded in an associative table of shared lock states (TSLS).

16. A method, as recited in claim 15,

wherein at least a subset of the shared lock states that represent locks owned by a single owner is encoded separately from the table of shared lock states (TSLS).

17. A method, as recited in claim 15,

wherein at least a frequently used subset of the shared lock states is encoded separately from the table of shared lock states (TSLS).

18. In a computational system wherein locks of identical value are represented as references to a shared lock state encoding of the value, a method of implementing a bulk delegation of locks from one or more delegator transactions to one or more delegatee transactions, the method comprising:

validating the bulk delegation based at least in part on ignore conflicts relationships amongst the delegatee transactions; and

if validated, performing the bulk delegation.

19. The method of claim 18,

wherein the performing of the bulk delegation includes, for each shared lock state encoding having an associated owner set that includes one of the delegator transactions, removing from the owner set each of the delegator transactions and adding thereto each of the delegatee transactions.

20. The method of claim 18,

wherein the validating is based on ignore conflicts relationships amongst the delegatee transactions and between the delegatee transactions and

otherwise incompatible-mode owners that would remain after completion of the bulk delegation.

21. The method of claim 18,

wherein the validating is based on one or more conservative validation tests that, if successful, obviate validity testing for each shared lock state; and

wherein, if not obviated by the one or more conservative validation tests, the validating includes, for each shared lock state encoding having an associated owner set that includes at least one of the delegator transactions, testing validity of delegating a lock represented thereby based, at least in part, on ignore conflicts relationships amongst the delegatee transactions and between the delegatee transactions and otherwise incompatible-mode owners that would remain after completion of the bulk delegation.

22. A transaction processing system that supports bulk delegation of locks, the transaction processing system comprising:

a lock manager that associates locking capabilities with transactions and that allows specification of certain conflicts between locking capabilities to be ignored; and

an encoding of ignore conflicts relationships amongst the transactions, the lock manager implementing at least a subset of bulk delegation operations by first validating requests therefor based on encoded ignore conflicts relationships amongst the delegatee transactions and between the delegatee transactions and otherwise incompatible-mode owners that which would remain after completion of the bulk delegation, and if the requests are validated by performing the bulk delegation operations.

23. The transaction processing system of claim 22,

wherein the transaction processing system implements a transaction model that supports a nested conflict serializability correctness criterion; and

wherein the bulk delegation operation is employed to support creation, commitment or abortion of a sub-database.

24. The transaction processing system of claim 22, wherein as part of an initial computationally-efficient, but conservative, validation, the lock manager:

verifies that, for each delegator transaction and type of conflict between lock modes, a set of transactions with which a particular delegatee transaction can ignore conflicts is a super-set of the set of transactions with which the delegator transaction can ignore conflicts; and

if at least one to-be-delegated lock is owned in a write mode by at least one of the delegator transactions, further verifies that each of the delegatee transactions can ignore conflicts with each other.

25. A lock manager that implements a bulk delegation operation by validating a bulk delegation request based, at least in part, on ignore conflicts relationships amongst delegatee transactions.

26. The lock manager of claim 25, embodied as software executable in an transaction processing environment that associates locking capabilities with transactions and that allows specification of certain conflicts between locking capabilities to be ignored.

27. A computer program product encoded in one or more computer readable media and comprising:

definition of a data structure instantiable in memory to represent plural locks having identical lock values using a single shared lock state encoding;

lock manager instructions executable by a processor to associate locking capabilities with transactions, to specify certain conflicts between locking capabilities to be ignored, to manage the shared lock state encoding, and to implement a bulk delegation of locks from one or more delegator transactions to one or more delegatee transactions, wherein the lock

manager instructions validate a bulk delegation request based, at least in part, on ignore conflicts relationships amongst the delegatee transactions.

28. The computer program product of claim 27,
wherein, if the bulk delegation request is successfully validated, the bulk delegation is performed by the lock manager.

29. The computer program product of claim 27,
wherein the one or more computer readable media are selected from the set of a disk, tape or other magnetic, optical or electronic storage medium and a network, wireline, wireless or other communications medium.

30. An apparatus comprising:
means for representing plural locks having identical values using a single shared lock state encoding; and
means for validating a bulk delegation of locks from one or more delegator transactions to one or more delegatee transactions based, at least in part, on ignore conflicts relationships amongst the delegatee transactions.